**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软工2405

学 号： 8209240529

姓 名： 熊俊冲

**实验报告成绩：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验** | **实验一** | **实验二** | **实验三** | **实验四** | **实验五** | **总评** |
| **成绩** |  |  |  |  |  |  |

**批阅教师：**

**实验三 函数**

**一、实验目的**

本实验主要培养、训练学生对函数的理解，要求：

1. 掌握函数的定义、声明的方法；

2. 掌握函数的编写要求；

3. 掌握函数的调用方法；

4. 掌握函数参数的传递方法；

5. 掌握变量的作用域；

6. 掌握多文件编程方法。

**二、实验内容与要求**

1、输入自然数m和n，

（1）求他们的最大公约数（或称最大公因数）。

要求输入、输出在主函数中进行，求公约数由函数实现。

#include<iostream>

using namespace std;

int gys(int a, int b)

{

if (a % b == 0)

{

return b;

}

else

{

return gys(b, a % b);

}

}

int main()

{

int a, b;

cin >> a >> b;

cout << "最大公约数为：" << gys(a, b);

}

99

15

最大公约数为：3

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 16136)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

1. 在函数中求最大公约数与最小公倍数。（提示：使用引用参数）

#include<iostream>

using namespace std;

int main()

{

int a, b, m;

cin >> a >> b;

for (int i = 0; a % b != 0; i++)

{

m = b;

b = a % b;

a = m;

}

cout << b;

}

99

15

3

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 36612)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

2. 编写程序满足：声明一个函数，判断一个整数是否为素数，使用如下函数头：

bool is\_prime(int num) ,如果num是素数函数返回true，否则返回false；

利用函数is\_prime找出前200个素数，并按每行10个输出：

     2     3      5      7    11    13    17    19    23    29

#include<iostream>

using namespace std;

bool is\_prime(int num)

{

for (int i= num-1; i>1; i--)

{

if (num % i == 0)

{

return 0;

}

}

return 1;

}

int main()

{

int a[200];

int n = 1;

for (int i = 2; n<201; i++)

{

if (is\_prime(i) == 1)

{

a[n-1] = i;

n++;

}

else

{

continue;

}

}

for (int i = 0; i < 200; i++)

{

if ((i+1) % 10 == 0)

{

cout << a[i] << endl;

}

else

{

cout << a[i]<<"\t";

}

}

}

2 3 5 7 11 13 17 19 23 29

31 37 41 43 47 53 59 61 67 71

73 79 83 89 97 101 103 107 109 113

127 131 137 139 149 151 157 163 167 173

179 181 191 193 197 199 211 223 227 229

233 239 241 251 257 263 269 271 277 281

283 293 307 311 313 317 331 337 347 349

353 359 367 373 379 383 389 397 401 409

419 421 431 433 439 443 449 457 461 463

467 479 487 491 499 503 509 521 523 541

547 557 563 569 571 577 587 593 599 601

607 613 617 619 631 641 643 647 653 659

661 673 677 683 691 701 709 719 727 733

739 743 751 757 761 769 773 787 797 809

811 821 823 827 829 839 853 857 859 863

877 881 883 887 907 911 919 929 937 941

947 953 967 971 977 983 991 997 1009 1013

1019 1021 1031 1033 1039 1049 1051 1061 1063 1069

1087 1091 1093 1097 1103 1109 1117 1123 1129 1151

1153 1163 1171 1181 1187 1193 1201 1213 1217 1223

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 15920)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

3、编程实现摄氏温度到华氏温度的转换：

编写一个头文件，包含下面两个函数：

double celsius\_to\_fah(double cel)    //摄氏温度到华氏温度

double fahrenheit\_to\_cels(double fah) //华氏温度到摄氏温度

实现头文件，并编写测试程序，调用函数显示如下结果：

Celsius    Fahrenheit   |   Fahrenheit       Celsius

40.0       105.0        |   120.0            48.89

39.0       102.0        |   110.0            43.33

……       ……        |   ……             ……

31.0        87.8        |   30.0             -1.11

（测试程序为主模块，即main( )函数所在的CPP文件，头文件mytemperature.h只有函数声明；函数定义写在另一CPP文件mytemperature.cpp）

4、创建名为mytriangle.h的头文件，包括：

bool is\_valid(double side1,double side2,double side3)

double\_area(double side1,double side2, double side3)

面积=sqrt(s(s-side1)(s-side2)(s-side3))

其中s=(side1+side2+side3)/2

写测试程序：读取三角形三边长，如输入合法，计算面积，否则输出错误信息。

（测试程序为主模块，即main( )函数所在的CPP文件，头文件mytriangle.h只有函数声明；函数定义写在另一CPP文件mytriangle.cpp）

#ifndef MYTRIANGLE\_H

#define MYTRIANGLE\_H

#include <iostream>

#include <cmath>

bool is\_valid(double side1, double side2, double side3);

double area(double side1, double side2, double side3);

#endif

#include "mytriangle.h"

bool is\_valid(double side1, double side2, double side3) {

if (side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 > side1) {

return true;

}

return false;

}

double area(double side1, double side2, double side3) {

double s = (side1 + side2 + side3) / 2;

return std::sqrt(s \* (s - side1) \* (s - side2) \* (s - side3));

}

#include "mytriangle.h"

int main() {

double side1, side2, side3;

std::cout << "请输入三角形第一条边的长度：";

std::cin >> side1;

std::cout << "请输入三角形第二条边的长度：";

std::cin >> side2;

std::cout << "请输入三角形第三条边的长度：";

std::cin >> side3;

if (is\_valid(side1, side2, side3)) {

double triangle\_area = area(side1, side2, side3);

std::cout << "该三角形的面积为：" << triangle\_area << std::endl;

}

else {

std::cout << "输入的三条边不能构成三角形，请重新输入。" << std::endl;

}

return 0;

请输入三角形第一条边的长度：5

请输入三角形第二条边的长度：4

请输入三角形第三条边的长度：3

该三角形的面积为：6

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 34068)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

**3与4选一个完成**

5、猴子吃桃：猴子第一天摘若干桃子，当即吃了一半，还不过瘾，又吃了一个。第二天又将剩下的桃子吃掉一半，又多吃一个，以后每天如此，到第10天，发现只剩最后一个桃子，问，第一天猴子共摘多少桃子（用递归实现）。

**三、实验思考题**

1. 本实验中函数中返回的值为什么与函数类型一致？

2. 本实验中主函数调用函数时采用的是何种传递方式？

**四、算法分析，程序结果**

**五、遇到的问题与解决方法**

**六、体会**

**实验四 数组与指针**

【**实验目的**】

1、进一步加深对数组的理解，掌握数组的定义方法；

2、掌握数组的处理方法、数组作为函数参数的使用方法，以及搜索与排序的应用。

3、掌握指针的概念、指针变量定义格式以及指针的运算；

4、掌握指针与数组、函数的关系；

5、理解内存动态分配的含义、熟练掌握内存动态分配方法；

6、掌握递归函数的定义方法。

【实验内容与步骤】

**（一）数组**

1、打印不同的数：

编写一个程序，读入10个数，输出其中不同的数（即如果一个数出现多次，只打印一次）。

提示：读入的数如果是一个新的值，则将其存入一个数组。否则，将其丢弃。输入完毕后，数组中保存的就是不同的数。

下面是一个运行样例：

Enter

Enter ten numbers: 1 2 3 2 1 6 3 4 5 2

The distinct numbers are: 1 2 3 6 4 5

#include<iostream>

int main()

{

int a[10];

for (int i = 0; i < 10; i++)

{

std::cin >> a[i];

}

int b[10] = { 0 };

int m = 0;

for (int i = 0; i < 10; i++)

{

bool n = false;

for (int j = 0; j < i; j++)

{

if (a[i] == a[j])

{

n = true;

break;

}

}

if (!n)

{

b[m] = a[i];

m++;

std::cout << b[m - 1] << "\t";

}

}

}

1

2

23

424

23

12

1

2

42

2

1 2 23 424 12 42

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 26372)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

2、起泡排序：

利用起泡排序算法编写一个排序函数。起泡排序算法分若干趟对数组进行处理。每趟处理中，对相邻元素进行比较。若为降序，则交换；否则，保持原顺序。此技术被称为起泡排序（bubble sort）或下沉排序（sinking sort），因为较小的值逐渐地“冒泡”到上部，而较大值逐渐下沉到底部。

算法可描述如下：

bool changed = true;

do

{

changed = false;

for (int j = 0; j < listSize – 1; j++)

if (list[j] > list[j+1])

{

swap list[j] with list[j+1];

changed = true;

}

} while (changed);

很明显，循环结束后，列表变为升序。容易证明do循环最多执行listSize – 1次。

编写测试程序，读入一个含有10个双精度数字的数组，调用函数并显示排列后的数字。

#include<iostream>

void swap(double& refa, double& refb)

{

int t;

t = refa;

refa = refb;

refb = t;

}

int main()

{

const int num = 10;

double a[num];

for (int i = 0; i < num; i++)

{

std::cin >> a[i];

}

bool changed = true;

do

{

changed = false;

for (int j = 0; j < (num-1); j++)

if (a[j] > a[j + 1])

{

swap(a[j], a[j + 1]);

changed = true;

}

} while (changed);

for (int i = 0; i < num; i++)

{

std::cout<< a[i]<<",";

}

}

23.2

23.11

2.2

1313.1

23.221

113.2

231.1

2.2

34.2

23.1

2.2,2.2,23,23,23,23.1,34,113,231,1313,

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 8852)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

3、游戏：存物柜问题：

一个学校有100个存物柜，100个学生。开学第一天所有存物柜都是关闭的。第一个学生（记为S1）来到学校后，打开所有的存物柜。第二个学生S2，从第二个存物柜（记为L2）开始，每隔两个存物柜，将它们关闭。第三个学生S3从第三个存物柜L3开始，每隔三个，将它们的状态改变（开着的关上，关着的打开）。学生S4，从L4开始，每隔四个改变它们的状态。学生S5，从L5开始，每隔五个改变状态。依此类推，直至学生S100改变L100的状态。

当所有学生完成这个过程，那些存物柜是开着的？编写一个程序求解此问题，显示所有开着的柜子号码，号码之间用一个空格隔开。

提示：使用一个100个布尔型元素的数组，每个元素代表存物柜是开（true）或关（false）。最初所有的储物柜都是关闭的。

#include <iostream>

int main()

{

bool lockers[100] = { false };

for (int student = 1; student <= 100; student++)

{

for (int locker = student - 1; locker < 100; locker += student)

{

lockers[locker] = !lockers[locker];

}

}

std::cout << "开着的存物柜号码为：";

for (int i = 0; i < 100; i++)

{

if (lockers[i])

{

std::cout << (i + 1) << " ";

}

}

std::cout << std::endl;

return 0;

}

开着的存物柜号码为：1 4 9 16 25 36 49 64 81 100

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 37028)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

4、合并两个排列好的数组：

编写如下函数，合并两个排列好的数组，形成一个新的排列好的数组。

void merge(const int list1[], int size1, const int list2[], int size2, int list3[])

使用size1+size2次比较实现函数。编写测试程序，提示用户输入两个排列好的数组，并显示合并以后的数组。下面是一个运行样例。注意，输入数据的第一个数字是数组的元素数，而不是数组的一部分。假定数组大小不超过80。

Enter

Enter

Enter list1: 5 1 5 16 61 111

Enter list1: 4 2 4 5 6

The merged list is 1 2 4 5 5 6 16 61 111

#include<iostream>

void merge(int size1, const int list1[], int size2, const int list2[], int list3[])

{

int i = 0, j = 0, k = 0;

while (k < size1 + size2)

{

if (i < size1 && (j >= size2 || list1[i] < list2[j]))

{

list3[k] = list1[i];

i++;

}

else

{

list3[k] = list2[j];

j++;

}

k++;

}

}

int main() {

const int MAX\_SIZE = 80;

int list1[MAX\_SIZE], list2[MAX\_SIZE], list3[MAX\_SIZE \* 2];

int size1, size2;

std::cout << "输入第一个已排序数组的元素个数：";

std::cin >> size1;

std::cout << "输入第一个已排序数组的元素：";

for (int i = 0; i < size1; i++) {

std::cin >> list1[i];

}

std::cout << "输入第二个已排序数组的元素个数：";

std::cin >> size2;

std::cout << "输入第二个已排序数组的元素：";

for (int i = 0; i < size2; i++) {

std::cin >> list2[i];

}

merge(size1, list1, size2, list2, list3);

std::cout << "合并后的数组为：";

for (int i = 0; i < size1 + size2; i++) {

std::cout << list3[i] << " ";

}

std::cout << std::endl;

return 0;

}

输入第一个已排序数组的元素个数：3

输入第一个已排序数组的元素：11

22

33

输入第二个已排序数组的元素个数：4

输入第二个已排序数组的元素：1

2

3

44

合并后的数组为：1 2 3 11 22 33 44

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 19872)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

5、检验子串：

编写如下函数，检验C字符串s1是否是C字符串s2的子串。如果匹配，返回s1在s2中的下标，否则返回–1。

int indexOf(const char s1[], const char s2[])

编写测试程序，读入两个C字符串，检验C字符串s1是否是C字符串s2的子串。下面是程序的运行样例：

Enter

Enter

Enter

Enter the first string: welcome

Enter the second string: We welcome you!

indexOf(“welcome”, “We welcome you!”) is 3

Enter

Enter the first string: welcome

Enter the second string: We invite you!

indexOf(“welcome”, “We invite you!”) is –1

#include <iostream>

#include <cstring>

int indexOf(const char s1[], const char s2[]) {

int s1Len = strlen(s1);

int s2Len = strlen(s2);

for (int i = 0; i <= s2Len - s1Len; i++) {

int j;

for (j = 0; j < s1Len; j++) {

if (s2[i + j] != s1[j]) {

break;

}

}

if (j == s1Len) {

return i;

}

}

return -1;

}

int main() {

char s1[100];

char s2[100];

std::cout << "请输入第一个字符串：";

std::cin.getline(s1, 100);

std::cout << "请输入第二个字符串：";

std::cin.getline(s2, 100);

int result = indexOf(s1, s2);

if (result != -1) {

std::cout << s1 << " 是 " << s2 << " 的子串，在 " << s2 << " 中的下标为：" << result << std::endl;

}

else {

std::cout << s1 << " 不是 " << s2 << " 的子串。" << std::endl;

}

return 0;

}

请输入第一个字符串：Hello

请输入第二个字符串：asdas Hello asdasd

Hello 是 asdas Hello asdasd 的子串，在 asdas Hello asdasd 中的下标为：6

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 20440)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

6、字符串中每个字母出现的次数：

请使用如下函数头编写函数，数出字符串中每个字母出现的次数。

void count(const char s[], int counts[])

counts是一个有26个元素的整数数组。const[0]，const[1]，…，const[25]分别记录a，b，…，z出现的次数。字母不分大小写，例如字母A和字母a都被看作a。

编写测试程序，读入字符串并调用count函数，显示非零的次数。下面是程序的一个运行样例：

Enter

Enter a string: Welcome to New York!

c: 1 times

e: 3 times

k: 1 times

l: 1 times

m: 1 times

n: 1 times

o: 3 times

r: 1 times

t: 1 times

w: 2 times

y: 1 times

#include<iostream>

#include <cstring>

#include <string>

void count(const char s[], int counts[])

{

int a = strlen(s);

for (int i = 0; i < a; i++)

{

for (int m = 65; m < 90; m++)

{

if ((s[i] == m) || (s[i] == (m + 32)))

{

counts[m - 65 ]++;

break;

}

else if (s[i] == 0)

{

continue;

}

else

{

continue;

}

}

}

}

int main()

{

std::string s;

std::getline(std::cin, s);

int alpha[26];

int counts[26] = { 0 };

const char\* ss = s.c\_str();

for (int i = 0; i < 26; i++)

{

alpha[i] = 65 + i;

}

count(ss, counts);

for (int i = 0; i < 26; i++)

{

std::cout << char(alpha[i]) <<"有" << counts[i]<<"个"<<std::endl;

}

}

Hellso Hsdhh as sadasda

A有4个

B有0个

C有0个

D有3个

E有1个

F有0个

G有0个

H有4个

I有0个

J有0个

K有0个

L有2个

M有0个

N有0个

O有1个

P有0个

Q有0个

R有0个

S有5个

T有0个

U有0个

V有0个

W有0个

X有0个

Y有0个

Z有0个

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 30924)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

**（二）指针**

1、上机验证下列程序的运行结果（有错误的话自己补充完善）

(1) void main()

｛

int i,j,\*pi,\*pj; //此处的\*表示定义指针变量，而非间接运算符

pi=&i;

pj=&j;

i=5;j=7;

cout<<i<<’\t’<<j<<’\t’<<pi<<’\t’<<pj;

cout<<&i<<’\t’<<\*&i<<’\t’<<&j<<’\t’<<\*&j;

}

运行结果：5 27168 000000BBF1CFF9F4 000000BBF1CFFA14000000BBF1CFF9F4 5 000000BBF1CFFA14 7

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 1864)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

上述结果中，pi与&i,pj与&j是地址值，随编译程序而变化，不确定。

(2) int main() //C语言程序，要了解

{

int a[]={1,2,3};

int \*p,i;

p=a; //将数组a首地址送给p

for (i=0;i<3;i++)

printf("%d,%d,%d,%d\n",a[i],p[i],\*(p+i),\*(a+i)); //与cout功能差不多

}

运行结果：

1,1,1,1

2,2,2,2

3,3,3,3

通过这两道题目，希望学生掌握数组元素与指向数组的指针的不同。

a[i]表示数组中下标为i的元素。

a[i]←p[i]←\*(p+i)←\*(a+i)

a是数组名，表示数组首地址，(p+i)表示数组中第i个元素的地址，\*(p+i) 相当于a[i]。

(3)通过如下的问题理解递归函数的定义与调用（递归未讲，可以后做）

//#include “stdio.h”

void f(char \*st,int i)

{

st[i]=’\0’;

cout<<st; // printf(“%s\n”,st);

if (i>1) f(st,i-1);

}

void main()

{

char st[]=”abcd”;

f(st,4);

}

补充完整，运行时输出为\_abcdabcaba\_\_\_\_\_\_\_

(4)下面程序的主函数中能保证p[0]输出1，p[1]输出2吗？如何修改以保证之（提示：在函数f中使用new生成动态数组；在main中用delete释放。）

#include<iostream>

using namespace std;

int \*f()

{

int list[]={1,2,3,4};

return list;

}

void main()

{

int \*p=f();

cout<<p[0]<<endl;

cout<<p[1]<<endl;

}

2、程序设计

(1)编写函数检查字符串s1是否为字符串s2的子串，若是，返回第一次匹配的下标，否则返回-1。在主程序中输入字符串s1与s2，调用函数实现。

函数原型：int indexof(const char \*s1,const char \*s2);

#include <iostream>

#include <cstring>

int indexOf(const char s1[], const char s2[]) {

int s1Len = strlen(s1);

int s2Len = strlen(s2);

for (int i = 0; i <= s2Len - s1Len; i++) {

int j;

for (j = 0; j < s1Len; j++) {

if (s2[i + j] != s1[j]) {

break;

}

}

if (j == s1Len) {

return i;

}

}

return -1;

}

int main() {

char s1[100];

char s2[100];

std::cout << "请输入第一个字符串：";

std::cin.getline(s1, 100);

std::cout << "请输入第二个字符串：";

std::cin.getline(s2, 100);

int result = indexOf(s1, s2);

if (result != -1) {

std::cout << s1 << " 是 " << s2 << " 的子串，在 " << s2 << " 中的下标为：" << result << std::endl;

}

else {

std::cout << s1 << " 不是 " << s2 << " 的子串。" << std::endl;

}

return 0;

}

请输入第一个字符串：Hello

请输入第二个字符串：asdas Hello asdasd

Hello 是 asdas Hello asdasd 的子串，在 asdas Hello asdasd 中的下标为：6

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 20440)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

(2)编写一个函数将以字符串形式表示的一个16进制数转换为10进制数，并在主函数中测试。函数原型 int parseHex(const char \*const hexString);

如：调用函数 parseHex(“A5”);返回165

#include<iostream>

#include<cstdlib>

int parseHex(const char\* const hexString)

{

char\* endptr;

long decimalNumber = strtol(std::string(hexString).c\_str(), &endptr, 16);

if (\*endptr != '\0')

{

return 0;

}

else

{

return decimalNumber;

}

}

int main()

{

std::cout << parseHex("A5") << std::endl;

}

165

C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 13056)已退出，代码为 0 (0x0)。

按任意键关闭此窗口. . .

1. 主程序中建立一动态数组（使用new），数组元素及元素个数由键盘输入，动态调试观察指针及指针指向的内容；设计一个函数对数组由小到大排序；主程序中用指针方式输出数组元素；最后释放数组内存（delete）。

#include<iostream>

void swap(int& refa, int& refb)

{

int t;

t = refa;

refa = refb;

refb = t;

}

int main()

{

int i;

std::cout << "数组元素个数：" << std::endl;

std::cin >> i;

int\* ptr = new int[i];

for (int j = 0; j < i; j++)

{

std::cin >> ptr[j];

}

bool changed = true;

do

{

changed = false;

for (int j = 0; j < (i - 1); j++)

if (ptr[j] > ptr[j + 1])

{

swap(ptr[j], ptr[j + 1]);

changed = true;

}

}

while (changed);

for (int j = 0; j < i; j++)

{

std::cout << ptr[j] << ",";

}

delete[] ptr;

}

**数组元素个数：**

**4**

**213**

**12**

**244**

**22**

**12,22,213,244,**

**C:\Users\99223\Desktop\实验\实验2\实验2\x64\Debug\Project1.exe (进程 19444)已退出，代码为 0 (0x0)。**

**按任意键关闭此窗口. . .**

【完成实验报告】

**实验报告只要求写程序设计部分**

**三、算法分析，程序结果**

**四、遇到的问题与解决方法**

**五、体会**